Classification with Sums of Separable Functions

Jochen Garcke

Technische Universität Berlin

DFG Research Center MATHEON
Mathematics for key technologies

December 2010, NIPS Workshop Tensors, Kernels, and Machine Learning

- beginning with scattered data in high dimensions

$$D = \left\{ (\underline{x}^j, y^j) = (x_1^j, \cdots, x_d^j; y^j) \right\}_{j=1}^{N} \qquad \underline{x}^j \in [0,1]^d, \; y^j \in \{-1, 1\}$$

- re-construct underlying function $f(\underline{x})$ such that
  - $\text{sign}(f(\underline{x}^j)) = y^j$
  - f provides a reasonable prediction when evaluated at other $\underline{x}$
- we get with
  - suitable loss/cost function L to minimise misclassification count
  - Tikhonov-regularisation (to have well-posed problem)

$$R(f) \underset{f \in V}{\longrightarrow} \min \, !$$

with

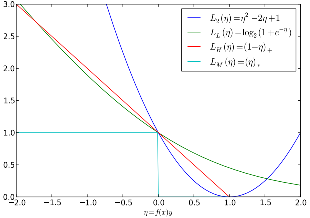
$$R(f) = \frac{1}{N} \sum_{j=1}^{N} L(f(\underline{x}^j), y^j) + \lambda \|\mathcal{S} f\|^2$$

- negative log likelihood

$$\frac{1}{N}\sum_{j=1}^{N} L_L(y^j, g(\underline{x}^j)) = \frac{1}{N}\sum_{j=1}^{N} \log\left(1 + \exp\left(-y^j g(\underline{x}^j)\right)\right)$$

- huberised hinge loss (h is a parameter to be chosen)

$$\frac{1}{N}\sum_{j=1}^{N} L_H(y^j, g(\underline{x}^j)); \quad L_H(y,t) = \begin{cases} 0 & \text{if} \quad yt > 1+h \\ \frac{(1+h-yt)^2}{4h} & \text{if} \quad |1-yt| \le h \\ 1-yt & \text{if} \quad yt < 1-h \end{cases}$$



- we employ a sum of separable functions

$$\boxed{f(\underline{x}) = \sum_{l=1}^{r} \prod_{i=1}^{d} f_i^l(x_i)}$$

- costs rdM if each (one-dimensional) f_i^l costs M
- good approximation with small r defeats curse of dimensionality
- represent $f_i^l \in V_i$ by its coefficients $\underline{c}_i^l$ for basis $\{\phi_k\}_{k=1}^M$

$$f_i^l = \sum_{k=1}^{M} c_i^l(k)\phi_k$$

- very closely related to low rank decomposition for tensors
- therefore in two dimensions very closely related to SVD
- Regression in [Beylkin.Garcke.Mohlenkamp:2009]

What is one-D f_i^l?

- we represent $f_i^l \in V_i$ by its coefficients $\underline{c}_i^l$

$$f_i^l = \sum_{k=1}^{M} c_i^l(k)\phi_k$$

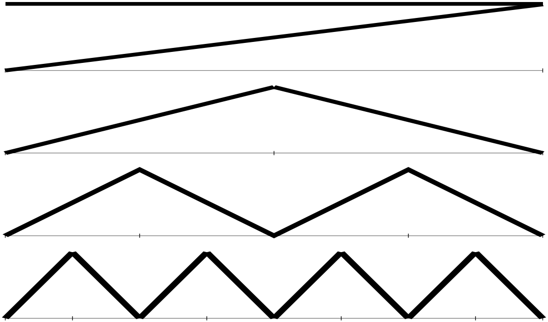
- use hat functions (piecewise linear), shown level 3, i.e. $M = 2^3 + 1$

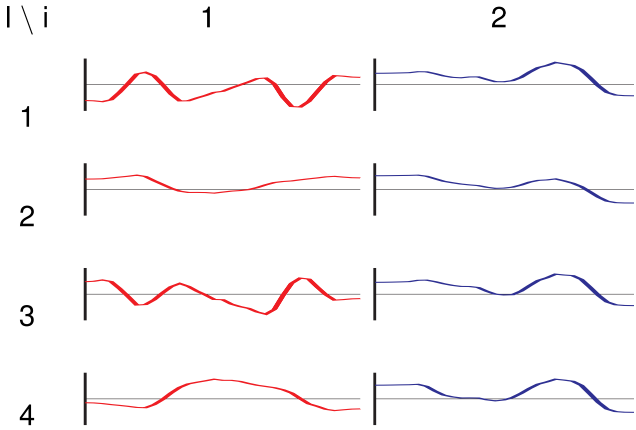
$0 : k = 0, 1$

$1 : k = 2$

$2 : k = 3, 4$

$3 : k = 5, 6, 7, 8$





$r = 4$, multi-scale basis with level 5, $M = 2^5 + 1$

Learning Theory Context

- framework of Sobolev spaces for learning theory
- bounds on its properties for learning e.g. in [Cucker.Smale:2001]
- discrete approximation takes place (by sums of separable functions)
- approximation theory bounds for convergence rates in statistical learning theory context in [Barron.Cohen.Dahmen.Devore:2008]
- other approaches can be formulated as sum of separable functions
  - with increasing rank r and resolution M one can approximate a function from a Sobolev space of certain smoothness arbitrarily close
  - convergence order for related approaches grows exponentially in d
- currently no characterisation of functions with low separation rank

Minimisation

- primary goal: investigate performance of function representation
- non-quadratic loss function need non-linear solution process
- essentially two strategies in this setting
  - minimise in whole parameter space (empirically does not work)
  - alternatingly minimise a subset of the unknowns at each step
- need non-linear minimisation for both, e.g.
  - BFGS Quasi-Newton
  - non-linear CG
  - trust-region Newton
- can hit local minima in any case

- loop over the dimensions $i = 1, \ldots, d$
  - fix the components in all directions but i

$$\text{e.g. } i = 1 : \quad f(\underline{x}^j) = \sum_{l=1}^{r} f_1^l(x_1^j) \prod_{i=2}^{d} f_i^l(x_i^j) = \sum_{l=1}^{r} f_1^l(x_1^j) p_j^l$$

  - improve f by modifying the components in one direction i
- L_L in one dimension

$$\frac{1}{N} \sum_{j=1}^{N} \log\left(1 + \exp\left(-y_j \sum_{l=1}^{r} s_l p_j^l f_1^l(x_1^j)\right)\right)$$

  - $\mathcal{O}(rMN)$ to compute loss function (& gradient)
- two variants for regularisation
  - ∇dD: use $\|\nabla f(\underline{x})\|^2$ to regularise
    - inner iteration with complexity $\mathcal{O}((rMN + r^2M^2)S + d^2r^2M^2)$
  - ∇f_i^l: regularise each $f_1^l(x)$ with $\|\nabla(f_1^l(x))\|^2$
    - inner iteration with complexity $\mathcal{O}((rMN + r^2M^2)S)$

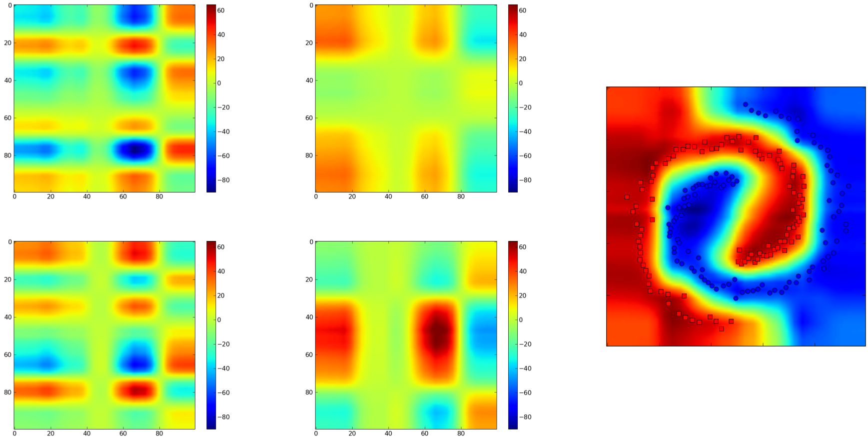
Total Cost

- in addition to the inner solver we have an outer iteration
  - main cost is update of

$$p_j^l = \prod_{i=2}^{d} f_i^l(x_i^j), \quad j = 1, \ldots, N, \quad l = 1, \ldots, r$$

  - one update can be done in $\mathcal{O}(rMN)$
  - therefore cost for one iteration is $\mathcal{O}(drMN)$
- with K the number of outer iterations we get complexity
  - $\mathcal{O}(Kd[(r^2M^2 + rMN)S + d^2r^2M^2])$ for ∇dD regularisation
  - $\mathcal{O}(Kd[(rMN + r^2M^2)S])$ for ∇f_i^l regularisation
- again complexity linear in N
- linear in d for ∇f_i^l regularisation

Ranks for Spiral Data Set Example



$r = 4$, multi-scale basis level $M_l = 5$

Empirical Comparison of Different Variants

data set	L_L Reg.∇f_i^l	L_H Reg.∇f_i^l	L_L Reg.∇dD	L_H Reg.∇dD
CIRCLE	2.00	2.10	1.95	2.30
SPIRALS	0.90	1.10	0.20	0.75
TWONORM	3.40	3.85	5.50	5.90
THREENORM	18.95	19.10	14.40	15.40
RINGNORM	4.80	4.90	4.70	5.30
CANCER	2.94	2.92	2.92	2.94
LIVER	25.71	25.71	30.56	30.56
CREDIT	22.77	24.25	26.87	26.73
IONOSPHERE	8.57	8.57	8.57	8.57
DIABETIS	22.08	23.23	22.72	23.38

- repeat procedure from benchmark study, 100 runs for each data set
- L_L (here) outperforms L_H
- "dirty" regularisation ∇f_i^l better for real data sets
- ∇dD better for synthetic data

Tests on Classification Datasets (17 Algorithms)

data set	L_L Reg.∇f_i^l	L_H Reg.∇f_i^l	L_L Reg.∇dD	L_H Reg.∇dD
CIRCLE	1	1	1	1
SPIRALS	3	3	2	3
TWONORM	5	5	10	11
THREENORM	8	9	2	2
RINGNORM	2	2	2	2
CANCER	4	3	3	4
LIVER	1	1	6	6
CREDIT	1	10	13	12
IONOSPHERE	4	4	4	4
DIABETIS	1	5	5	6

- position in comparison to benchmark study with 17 algorithms
- for eight of the data sets in the top three
- for six data sets at least one version achieved better results than svm
- not more than 7 ranks used, mostly less

- competetive results for classification with sums of sep. functions
- often surprisingly low ranks to describe classifier
- computational cost can be as low as $\mathcal{O}(Kd[(rMN + r^2M^2)S])$
- there are variations and possible extensions e.g.
  - non-negative functions for better interpretability in case of L_L
  - multi-class loss functions for hinge loss or penalized likelihood estimation using vector-valued functions
  - different one-dimensional spaces for different attributes and ranks
- more sophisticated minimisation strategies needed

4th Workshop on High-Dimensional Approximation

HDA2011 — June 26–30, 2011 — University of Bonn, Germany

About

The workshop covers current research on all numerical aspects of high-dimensional problems. The scope ranges from high-dimensional approximation theory over computational methods to engineering and scientific applications. Participation is open to all interested in high-dimensional computational mathematics and science.

This international workshop is the fourth in a series which were previously held at *The Australian National University* in Canberra (HDA05 and HDA07) and at the *University of New South Wales* in Sydney (HDA09). This year the workshop takes place at the *University of Bonn*. It is embedded in the *Hausdorff Trimester Program on Analysis and Numerics for High-Dimensional Problems*.

Scientific committee

- Jochen Garcke (Matheon & TU Berlin, Germany)
- Michael Griebel (U Bonn & Fraunhofer SCAI, Germany)
- Max Gunzburger (Florida State U, USA)
- Wolfgang Hackbusch (Max Planck Institute Leipzig, Germany)
- Markus Hegland (ANU, Australia)
- Frances Kuo (UNSW, Australia)
- Christoph Schwab (ETH Zürich, Switzerland)
- Ian Sloan (UNSW, Australia)
- Henryk Wozniakowski (Columbia U, USA & Warsaw U, Poland)
- Harry Yserentant (TU Berlin, Germany)

Organisers

- Dirk Nuyens (KU Leuven, Belgium)
- Christian Rieger (U Bonn, Germany)

http://hda2011.ins.uni-bonn.de